Enhanced Facial Detection

In this section, I will explain what I’ve done to improve the tilted face detection hit rate. But before that, I will briefly go through how basic OpenCV’s facial detection with haar cascade works.

**First thing first, what is Haar cascade?**

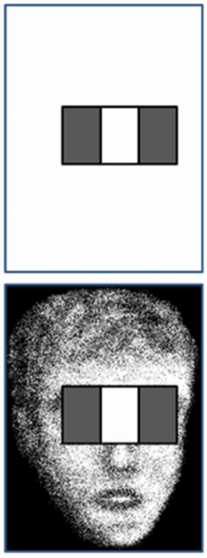
It is an object detection algorithm that makes use of the edge/line detection features proposed by Viola and Jones in their research paper “Rapid Object Detection using a Boosted Cascade of Simple Features” published in 2001. Or to my understanding, it’s an algorithm that uses object model that consists of features (often those features are selected using supervised machine learning-based approach) that best describe the model itself.

For example, if we want to detect a face, some of the features that might be selected are eye browns, eyes, nose, etc... But how do we represent those features? Or how can a computer “see” those features?

Say we have a forehead feature. The forehead is above the eyes, and usually the forehead is brighter than the eyes. So, we can convert this characteristic into an edge feature (reference the image below).

  
When scanning for faces in an image, for every potential face, the program will go to the position of the edge/line features, and check if the area covered by that edge/line satisfies the feature’ conditions. For this case, we have an edge feature that represents the forehead. If the added-up color value of the white box is brighter than the added-up color value of the black box, that means the program “see” a forehead.

Similarly, given a pair of eyes feature, because in between the eyes its usually lighter than the eyes, it can be represented as a line feature.



All these haar features are then grouped together into an object model to represent the characteristics of that object. The computer can just use that object when scanning. If all the haar features are present, then the object is “visible” to the computer. If not, then the computer “sees” nothing.

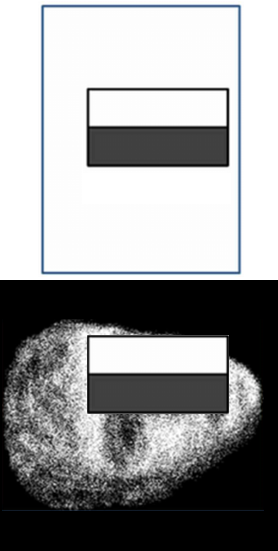


To summarize, OpenCV obtain a set of Haar features that best describe a face (provided by the user) and use that as a reference when checking for faces. OpenCV go through the image and crop out rectangles (potential face) in all possible different sizes and in all possible locations, compare those cropped out areas with the reference face to check if those satisfy all the facial feature requirements. If any of them do, then that will be considered a detected face.

**Limitation**

Because all the haar features are placed relatively to the potential face bounding box. If the box is not aligned with the angle of the face in the image, the all the haar features will be misaligned. Therefore, OpenCV would not be able to detect tilted faces.

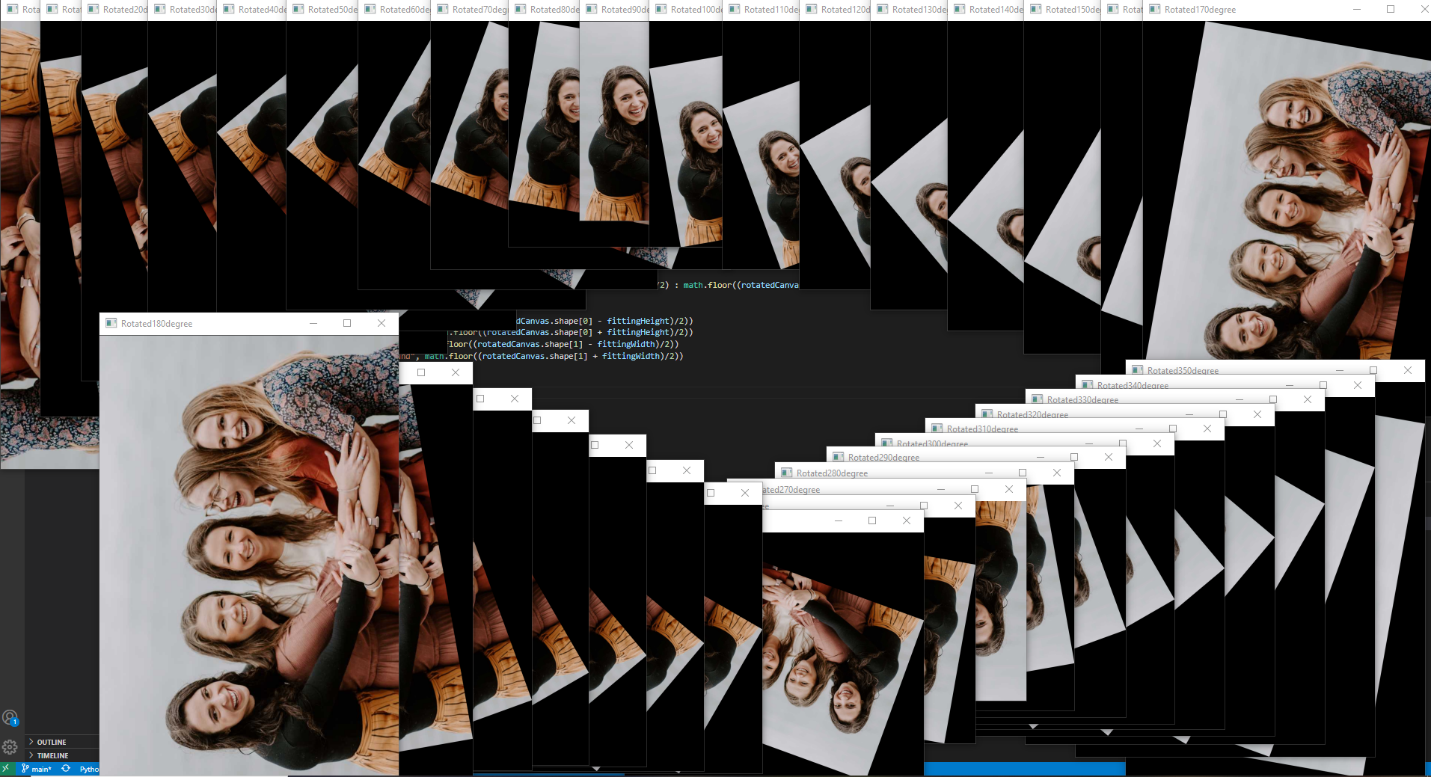
An example of the forehead features but now it is being used on a horizontal face



To combat this, my solution is to just simply do the scan in multiple orientations, not just on the vertical image. Once those scans finish, I can just convert the rotated coordinates of the faces found back to the original image’s coordinates axis, check if there is any overlapping face and remove the duplicates.

One small optimization that I made was to instead of scanning for faces directly, the program scanned for eyes (which takes half the time compared to scanning for faces). Then it checks if the eyes have similar size and the distance between the two is appropriate (not too close/too far). For every pair of eyes it finds, it will crop out the potential face area around that pair of eyes, then do facial detection on that small area only. If a face is found, I can use the relative locations of the eyes to further investigate which one is the left eye, which one is the right eye.

-My rotation function in action



-A glimpse into what my program sees for each step

